NEUROMORPHIC LEARNING OF CONTINUOUS-VALUED MAPPINGS IN THE PRESENCE OF NOISE:
Application To Real-Time Adaptive Control

Terry Troudet
Sverdrup Technology, Inc.
NASA Lewis Research Center Group
Cleveland, Ohio

Walter C. Merrill
National Aeronautics and Space Administration
Lewis Research Center
Cleveland, Ohio

Abstract

The ability of feed-forward neural net architectures to learn continuous-valued mappings in the presence of noise is demonstrated in relation to *parameter identification* and *real-time adaptive control* applications. Factors and parameters influencing the learning performance of such nets in the presence of noise are identified. Their effects are discussed through a computer simulation of the Back-Error-Propagation algorithm by taking the example of the cart-pole system controlled by a non-linear control law. Adequate sampling of the state space is found to be essential for canceling the effect of the statistical fluctuations and allowing learning to take place.

1 Introduction.

A major challenge for health-monitoring and diagnosis of complex Advanced Propulsion Systems (APS), such as the Space Shuttle Main Engine, is to perform real-time analysis of a massive amount of diverse sensor data. Such analysis can be used either to directly perform low-level, real-time adaptive control or to send real-time descriptions of the dynamical state of the APS to a high-level controller. In the first case, the low-level controller has to compute adaptively and in real-time the control signal to be applied to the controlled process for a given set of sensor data. In the second case, the diagnosis process consists of "translating" in real-time the sensor information into one or several parameters which characterize specific aspects of the dynamical evolution of the APS. (For a discrete mapping, parameter identification reduces to pattern recognition). The massive parallelism of neural nets [1] and their ability to learn complex continuous mappings [2] give them the real-time computational power needed for such low-level monitoring/diagnosis functions.

When a realistic model of physical phenomena and feedback mechanisms is possible, adaptive control laws and parameter identifiers can be expressed in analytical forms, and neural nets can be trained from computer-generated data. In most instances however, it is difficult to derive realistic models, and neural nets can only be trained on experimental data which have been corrupted by a certain amount of noise during the signal processing of the sensor measurements.

In this paper, we analyze the effect of noise on the neuromorphic abilities of feed-forward neural net architectures to learn continuous mappings, and to serve as *parameter identifiers* or *real-time adaptive controllers*. A discussion of a computer-simulation of the training architecture described in Section 2 is presented in Section 3 for a non-linear continuous-valued mapping by taking the example of a controlled cart-pole system. The training sequence is analyzed in detail with and without noise in the training data.

2 Training Architecture.

In the presence of noise, the state variables of the controlled process, $\vec{Z}$, and the applied control signal, C, are

$$\vec{Z} = \vec{Z}^{exact} + \hat{n}_{\vec{Z}} \qquad (1)$$

and

$$C = C^{exact} + \hat{n}_C \qquad (2)$$

where the noises $\hat{n}_{\vec{Z}}$ and $\hat{n}_C$ are simulated as independent variables, normally distributed with zero mean. For a high signal-to-noise ratio of the input signal, i.e. $\frac{max||\vec{Z}^{exact}||}{\sqrt{var(\hat{n}_{\vec{Z}})}} >> 1$, the function ϕ which represents the control law, $\phi(\vec{Z}^{exact}) = C^{exact}$, can be expressed, using a Taylor expansion, as

$$\phi(\vec{Z}) \approx C^{exact} + \hat{n}_{\vec{Z}} \frac{\partial \phi}{\partial \vec{Z}}_{(\vec{Z}=\vec{Z}^{exact})} \quad (3)$$

In this case, trying to learn the mapping ϕ from sets of noisy input/output vectors $(\vec{Z}, C)$ is thus equivalent to trying to learn ϕ from sets of input/output vectors $(\vec{Z}^{exact}, C_n)$ where only the control force is corrupted by noise:

$$C_n = C^{exact} + \hat{n} \quad (4)$$

$$\hat{n} \approx \hat{n}_C - \hat{n}_{\vec{Z}} \frac{\partial \phi}{\partial \vec{Z}}_{(\vec{Z}=\vec{Z}^{exact})} \quad (5)$$

This equivalence shows that the factors and parameters which influence neuromorphic learning with noisy input/output data can be studied by analyzing the learning performance of the net as a function of the signal-to-noise ratio $\frac{max|C^{exact}|}{\sqrt{var(\hat{n})}}$ of the output signal.

The first phase of the training consists of sampling at various times t_k the state variables $\vec{Z}^{exact}(t_k)$ of the controlled process, Fig.(1), and the control signal $C_n(t_k)$ given in Eqs.(4) and (5). In the second phase, the set of training data $(\vec{Z}^{exact}(t_k), C_n(t_k))$ is *organized* in I/O subsets before being applied to the neuromorphic controller.

3 Cart-Pole System.

In this example, the controlled process of Figure 1 is chosen to be the cart-pole system [3] - [4] represented in Figure 2. Training data are recorded by placing initially the cart-pole at arbitrary positions $(X(0), \theta(0))$ with zero velocities, and by driving it to the origin $(X = 0, \theta = 0)$ with a control force. While the cart-pole is returned to its equilibrium position, the four-dimensional state vector $\vec{Z}(t) = (X(t), \dot{X}(t), \theta(t), \dot{\theta}(t))$, and the control force, $F(t)$, are regularly sampled over a certain period of time. Sampling rate and observation time are considered as parameters of the training.

Non-Linear Control Law.

The dynamic evolution of the cart-pole with its parameters shown in Figure 2 is given by the equations of motion

$$\ddot{\theta} = h_1 - h_2 \ddot{X} \quad (6)$$

$$\ddot{X} = \frac{f_1 + F}{f_2} \quad (7)$$

where

$$h_1 = \frac{3}{4L} g sin\theta \quad (8)$$

$$h_2 = \frac{3}{4L} cos\theta \quad (9)$$

$$f_1 = m(L sin\theta \dot{\theta}^2 - \frac{3}{8} g sin2\theta) - f\dot{X} \quad (10)$$

$$f_2 = M + m(1 - \frac{3}{4} cos^2\theta) \quad (11)$$

The control force F is generated by application of a feedback linearizing and decoupling transform [5]

$$F = \frac{f_2}{h_2}[h_1 + k_1\theta + k_2\dot{\theta} + c_1 X + c_2 \dot{X}] - f_1 \quad (12)$$

with $k_1 = 25$, $k_2 = 10$, $c_1 = 1$ and $c_2 = 2.6$. Training data are generated by integrating the set of Eqs.(6)-(12) with the initial condition $\vec{Z}(0) = (X(0), 0, \theta(0), 0)$, $X(0)$ and $\theta(0)$ being arbitrary position and angle of the domain $D_{X\theta}$=[-4m,+4m] × [-50deg,+50deg]. The neural architecture chosen to approximate the control law $F[\vec{Z}]$ is a feed-forward net of 4 linear neurons in the input layer, two hidden layers of 16 and 4 neurons respectively, and one neuron in the output layer. The continuous values of $\vec{Z} = (X, \dot{X}, \theta, \dot{\theta})$ are fed into the input layer. Each neuron input is connected to all neuron outputs of the previous layer, and to a unit which is permanently on (threshold term). The output signal of each neuron is linearly modulated by the (synaptic) weights before exciting/inhibiting a connected neuron. With the standard activation function

$$output = \frac{1}{1 + e^{-input}} \quad (13)$$

the output o_n of a neuron is bound to the interval [0,1]. Restricting the information domain of a neuron to the interval [0.1,0.9] by eliminating the asymptotes of the input/output

response curve of Eq.(13) is known to enhance learning performance [6]. The neuromorphic learning of the continuous-valued mapping $F[\bar{Z}]$ requires the scaling and offsetting of the last layer output

$$u = \frac{F}{F_g} = 2.5 * o_n - 1.25 \qquad (14)$$

where F_g is a constant parameter which normalizes the control signal over [-1,+1]. It is essential to emphasize that the choice of F_g defines the domain where the mapping is to be learned, and influences the neural computation as well. Eqs.(13) and (14) show that the net output *cannot* match values of F such that $|F| > 1.25F_g$ (practically, the net can only match accurately the domain $|F| \leq F_g$). When a control law or a continuous-valued mapping is to be learned, it is imperative to first define the domain of variations of the signals before choosing the constant parameter F_g since it has to be larger than the absolute maximum value of the signal, $max|F|$. Among the possible values of F_g satisfying $F_g \geq max|F|$, the optimal value corresponds to the best approximation of the continuous-valued mapping by the neural net.

It is well-known that linear mappings - those for which the control signal is a linear function of the state variables - can be learned by a single-layer perceptron. In that case, it is easy to show that perfect learning occurs over any domain of variations of the signals when the limit $F_g \rightarrow \infty$ is taken. In the general case however, the only way to find the optimal F_g is to train the net for several values F_g larger than $max|F|$, and compare the learning performances. In this work, we have chosen $F_g = max|F|$ ad hoc.

The results hereafter have been obtained by training the net with the Back-Error-Propagation (BEP) algorithm on a VAX 8650 of the NASA Lewis VAXCLUSTER. The initial values of weights and thresholds have been chosen randomly distributed in the interval $[-0.1, +0.1]$ to break symmetries that could eventually lead to spurious modes and bias the learning.

3.1 Absence of Noise.

In order to extract the relational law between state variables and control force from a bulk of input/output data, it is necessary that the training data be as representative as possible of the state space, and that they be sampled as uniformly as possible throughout the state space. First, if there are too few training data or if they are not sufficiently representative, the neural net may either only "memorize" the training data without extracting the control law itself, or approximate the control law only over a partial domain. This means that the net may not be able to reproduce control signals corresponding to state variables that were not part of the training set. Second, if the sampling is not uniform throughout the state space, learning may be biased, and the net may only "memorize" part of the training data without extracting the features contained in the training set.

When the dynamic characteristics of the cart-pole are regularly sampled until it returns to the origin, the distribution of the training forces $\{u\}$ is strongly peaked around the origin. Clearly, a random sampling of the training data would bias the training and prevent the net from learning the control law for large displacements of the cart-pole. Before training, data are organized by dividing the interval [-1,+1] of the normalized force $u = \frac{F}{F_g}$ into 11 subintervals $\{I_k, k = 1, 11\}$ of equal length. One training iteration consists of the *random* sampling of a subinterval I_k followed by the *random* sampling of a $u \in I_k$. The corresponding set of state variables is used as input to the net. The resulting network output is compared to the target output $o_n = \frac{u+1.25}{2.5}$. Each individual error between target output and network output is propagated through all the net layers to update weights and thresholds by BEP. This process is iterated until convergence.

The accuracy of the neural approximation of the control law can be characterized by the total mean-square error e^2

$$e^2 = \frac{\sum_{k=1}^{11} e_k^2}{11} \qquad (15)$$

where e_k^2 is the mean-square error over the subinterval I_k:

$$e_k{}^2 = \frac{\sum_{i=1}^{n(k)} \left(u(i)^{net} - u(i)^{target}\right)^2}{n(k)} \qquad (16)$$

In Eq.(16), $u(i)^{target}$ is one of the $n(k)$ target data of I_k, and $u(i)^{net}$ the output of the net corresponding to the same state variables. The convergence of the algorithm can be estimated by evaluating the error e^2, Eq.(15), made by the net in trying to reproduce the data used for training; similarly, the degree of accuracy of the feature extraction can be estimated by evaluating the error e^2 made by the net in trying to predict the control forces corresponding to state vectors that were not presented to the net during training. When both errors are small and comparable in magnitude, the neural net has developed a good internal representation of the mapping.

It is well-known that any a-priori knowledge on the mapping features, such as symmetries, can efficiently improve the net performance when these are incorporated explictly in the neural computation. Since the control force changes as $F \rightarrow -F$ when the state vector changes as $\vec{Z} \rightarrow -\vec{Z}$, the ensemble of training data has been chosen symmetric under the transformation $T^- = [(\vec{Z}, F) \rightarrow (-\vec{Z}, -F)]$ by randomly distributing the initial position and angle of the cart-pole over the domain $D_{X\theta}$= [-4m,+4m] × [-50deg,+50deg].

Since all subintervals $\{I_k\}$ are treated with equal probability during training, there need to be enough training data to provide as uniform a representation of the control law over each I_k as possible. In this section, training was performed over 200 motions sampled at 20Hz over 10s. An upper-bound for the control force needed to bring the cart-pole back to the origin from a position in $D_{X\theta}$, and without initial velocities, is $max|F| = 120$Newtons. The BEP algorithm was run for $F_g = 120N$, with a steepest descent coefficient $\eta = 0.2$ and a momentum term $\alpha = 0.9$. Since the BEP algorithm is based on a minimization principle, there is always a possibility that the system of weights/thresholds may get *trapped* in a local minimum or may *freeze* in a flat spot during the search for the global minimum. In learning the mapping $F[\vec{Z}]$, Eq.(12), and using it to control the cart-pole, it was found that the neuromorphic controller was able to stabilize the pole to $\theta = 0$, but would occasionally return the cart to a position $X \neq 0$. In order to circumvent this difficulty due to the existence of a local minimum or a flat spot, we performed a fine-tuning by augmenting the training with data randomly sampled in the smaller subinterval I_θ which is centered around 0 and symmetric under T^-.

After training, the learning performance of the net can be tested in three different configurations. The "learning open-loop configuration" tests the ability of retrieving F from state variables $\vec{Z}$ used for training; the net is used as an *analog memory*. The "generalizing open-loop configuration" tests the ability of generating F from state variables not used during training; the net is used as a *parameter estimator*. The "generalizing closed-loop configuration" tests the ability of stabilizing the controlled process for a motion not used during training; the net is used as a *real-time adaptive controller* (RTAC).

The curves of Figures 3-4 show the result of the neural computation in the "generalizing" modes, after 99000 gross-tuning iterations and 1000 fine-tuning iterations. In the set of Figures 3, the initial state vector is $\vec{Z}(0) = (-1m, 0, 45deg, 0)$, and in Figures 4, $\vec{Z}(0) = (3m, 0, -35deg, 0)$.The neural net is able to return, very satisfactorily, the cart-pole to the origin from large angles and large displacements. This shows that, during training, the net has developed an *internal representation* which is a very good approximation of the mapping defined in Eq.(12).

With the above set of parameters, estimations of the error e^2, Eqs.(15)-(16), are 0.0004 in the "learning open-loop" mode, and 0.0006 in a "generalizing open-loop" mode for a set of 200 randomly generated cart-pole motions.

3.2 Presence of Noise.

With noise, the normalized values of the control forces used as targets are no longer the exact values since

$$u_n^{target} = u^{exact} + \hat{n} \qquad (17)$$

However, the mapping $F[\vec{Z}]$ can be learned by BEP from a training set of representative data $u_n^{target}(i)$. Given the statistically averaged square error function $< (F[\vec{Z}] + \hat{n}_F - G[\vec{Z}])^2 >$ over the entire state space, the BEP yields a function, $G(\vec{Z})$, which minimizes the error as shown variationally in Eqs.(18)-(19):

$$\frac{\delta < (F[\vec{Z}] + \hat{n}_F - G[\vec{Z}])^2 >}{\delta G[\vec{Z}]} = 0 \Longrightarrow \quad (18)$$

$$G[\vec{Z}] = F[\vec{Z}] \quad (19)$$

In the absence of noise, learning takes place only when the net is provided with many representative training data in order to be able to reproduce the features rather than "memorize" the data. This is even more crucial with noise since the target data are not the exact values of the force.

It is essential to choose small values for the steepest-descent parameter η in order to minimize the effect of the fluctuations. For a large η, samples with big deviation would overcontribute to the weights adjustments and mislead the search for the minimum. For a small η, the effect of such deviants tend to be balanced towards average by the contribution of the samples with small deviation since those occur with a higher probability. From a geometric point-of-view, the fluctuations drastically complicate the topology of the error-surfaces by creating more irregularities and introducing more possibilities for the net to get trapped in local minima or flat spots. Small values of η favor adiabatic changes of the weights towards paths of the energy surface which correspond to averaged values of the training data, i.e. $< u_n^{target} >=< u^{exact} + \hat{n} >= u^{exact}$. In the absence of noise, when the target values are the exact values, the role of the momentum term is to speed up the convergence process by amplifying the weight adjustments. In the presence of noise, a momentum term would amplify the undesired weight changes due to highly deviant data, and make adiabaticity more difficult to maintain. For this reason, the momentum coefficient is chosen $\alpha = 0$. The price to pay for a small steepest-descent coefficient and no momentum term is of course more iterations to reach convergence.

Fluctuations make it also difficult to generate representative sets of training data due to the *"contamination"* of the subintervals I_k. Due to noise, training data are likely to lie in subintervals that are different from the subintervals where the exact data are. For a non-uniform density distribution of the data used for training, this implies that the sampling by subintervals as described in Section 3.1 will be less uniform as the noise increases.

If data were obtained by regularly sampling cart-pole motions over "long" periods of time, e.g. 10 sec or more, the exact density distribution would be well peaked around 0. Noise would cause these data to spread from the center towards the edges of [-1,+1]. For a given noise intensity, the number of data spread out of a subinterval would be proportional to the number of data contained in that interval. As noise increases, data from subintervals located around the center, e.g. I_6, would increasingly populate the neighboring subintervals. For low signal-to-noise ratio $S/N = \frac{F_g}{\sqrt{var(\hat{n})}}$ and long observation time, the sampling method of Section 3.1 would most probably select noisy data originating from exact data located in I_6, I_5 or I_7, regardless of the chosen interval. In the limit, the fluctuations around the origin would be averaged out, and the net would essentially learn that $\vec{Z} = 0$ is the equilibrium position without being able to stabilize the cart-pole.

Such a "contamination" can only be restrained by as much as possible uniformly sampling the density distribution of the data before training. Since the exact values are not known, this can be done by shortening the observation time of the cart-pole motions and by increasing the number of these motions. Oversampling of similar dynamic states, e.g. the equilibrium position, would thus be prevented.

For the simulation, we have assumed that the mapping to be extracted and learned by the net is symmetric with respect to the T^- transformation, and we have chosen the initial positions/angles of the cart-pole randomly distributed over $D_{X\theta}$. Towards the end of the training, fine-tuning has been obtained by training the net with data only sampled from the subinterval I_6 which is smaller than [-1,+1] and still symmetric with respect to the T^- transformation. Training data have been generated by integrating the equations of motion (6)-(12), and adding a gaussian noise of zero mean. Samples of noisy cart-pole motions used for training are shown in the set of Figures 5a and 6a, and the results of the computation are shown in Figures 5b-e and 6b-e for $F_g = 120N$ with noise-to-signal ratios N/S=0.1 and N/S=0.2.

For N/S =0.1 (Figs 5b-e), training has been made by sampling 1000 cart-pole motions at

20 Hz over only 5 sec. The BEP parameters are $\eta = 0.02$ and $\alpha = 0$; 490,000 gross-tuning/10,000 fine-tuning iterations have been used. In the "learning open-loop" configuration, $e^2 = 0.006$, and in a "generalizing open-loop" configuration, $e^2 = 0.006$ for a set of 1000 randomly generated cart-pole motions.

In the RTAC configuration, the net itself controls the evolution of the cart-pole; the state vector $\vec{Z}(t)$ is therefore different from the state vector corresponding to the exact control law (12). This explains the discrepancy between Figs. 5b and 5c.

For N/S=0.2 (Figs.6b-e), 4000 cart-pole motions have been sampled at 20Hz over 2 sec only. The BEP parameters are $\eta = 0.01$ and $\alpha = 0$; 950,000 gross-tuning/50,000 fine-tuning iterations have been used. In the "learning open-loop" configuration, $e^2 = 0.008$, and in a "generalizing open-loop" configuration, $e^2 = 0.008$ for a set of 4000 randomly generated cart-pole motions.

These results show that, in spite of a significant amount of noise, the net is able to learn the control law (12) within sufficient accuracy to return satisfactorily the cart-pole to its equlibrium position.

In the presence of noise, an error function can be introduced to optimize the parameters of the computation and estimate the degree of accuracy to which the neural mapping approximates the exact unknown mapping. It is defined from Eqs.(15)-(16), but where $u(i)^{net}$ (expected to be closer to the exact value than $u(i)^{target}$ is) is one of the $n(k)$ net outputs contained in I_k, and $u(i)^{target}$ the corresponding noisy data. In the statistical limit where $n(k) \rightarrow \infty$ and for perfect learning, this mean-square error would be minimal and equal to the variance of the noise.

4 Conclusion.

These preliminary results demonstrate that neural networks can extract within a good aproximation a continuous-valued, non-linear mapping between input and output data when the training data are corrupted by noise. In order to extract the features contained in the training data, it is necessary to sample the state space as uniformly as possible. In the absence of noise, aproximately uniform samplings can be obtained by randomly sampling subintervals of the state space. However, the presence of noise mixes the populations of these subintervals, making it more difficult to reach an approximately uniform sampling throughout the state space. Limiting the observation time of each cart-pole motion, and increasing the number of such motions has been found to significantly reduce this contamination effect. It would be interesting to analyze the possibility of further reducing this contamination by pre-processing the training data with the help of neuromorphic classifiers, such as counter-propagation feed-forward nets, before training with BEP.

Acknowledgments.

We would like to thank Allon Guez and John Selinsky for stimulating discussions. We express our gratitude to Carl Lorenzo for his support during the realization of this work, and thank Jonathan Litt and Ten-Huei Guo for helping us clarify the manuscript.

References

[1] R. P. Lippmann: *"An Introduction To Computing With Neural Nets"*, IEEE ASSP Magazine, April 1987, pp.4-22.

[2] A. Guez and J. Selinsky: *"A Trainable Neuromorphic Controller"*, Journal of Robotics Systems, 5(4), 363-388 (1988).

[3] B. Widrow: *"The Original Adaptive Broom Balancer"*, IEEE Conf. on Circuits and Systems, Philadelphia, PA, 1987.

[4] A. G. Barto, R. S. Sutton, and C. W. Anderson: *" Neuronlike Adaptive Elements That Can Solve Difficult Learning Control Problems"*, IEEE Trans. on Systems, Man and Cybernetics, SMC-13(5), 834-846 (1983).

[5] A. Guez: *"Optimal Control Of Robotic Manipulators"*, Ph.D. Thesis, Univ. of Florida, 1983.

[6] D. Rumelhart, G. E. Hinton, and R. J. Williams: *"Learning Internal Representations by Error Propagation"*, in Parallel Distributed Processing, D. E. Rumelhart and J. L. McClelland, Eds., MIT Press (1986).

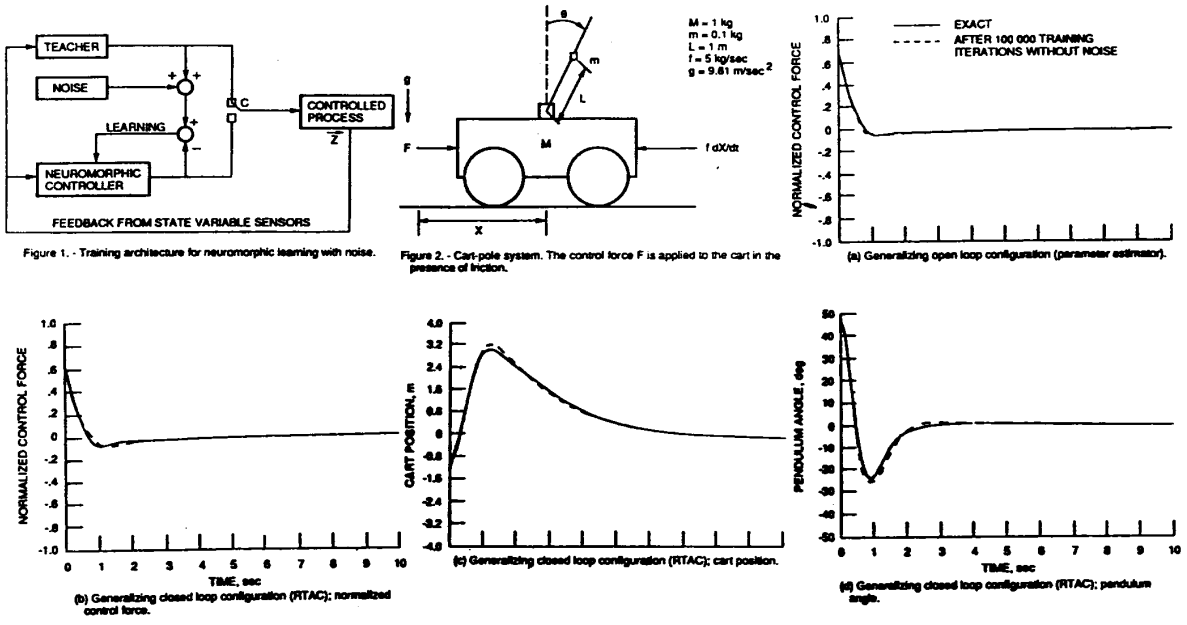


Figure 1. - Training architecture for neuromorphic learning with noise.

Figure 2. - Cart-pole system. The control force F is applied to the cart in the presence of friction.

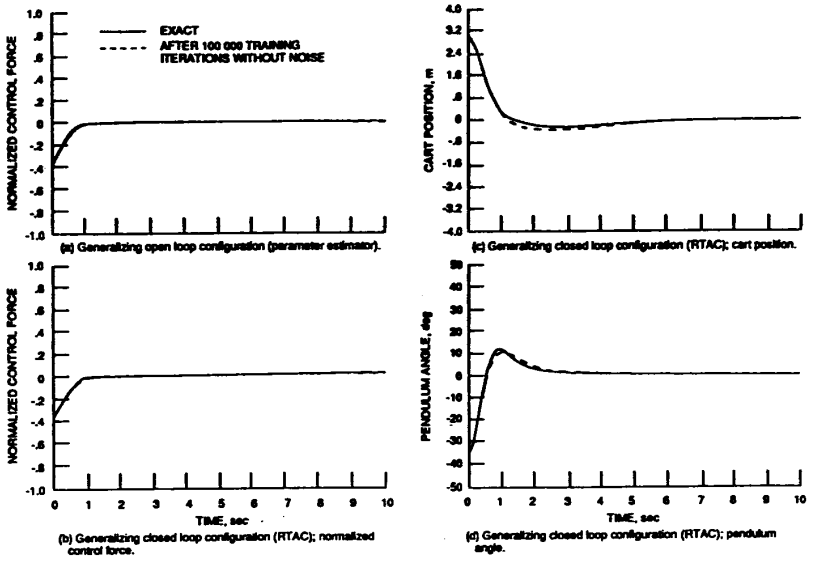
(a) Generalizing open loop configuration (parameter estimator).

(b) Generalizing closed loop configuration (RTAC); normalized control force.

(c) Generalizing closed loop configuration (RTAC); cart position.

(d) Generalizing closed loop configuration (RTAC); pendulum angle.

Figure 3. - Non-linear control law without noise. Initial state vector, $\vec{Z}(0)$ = (-1m, 0, 45 °, 0). Normalization constant, F_g =120 Newtons. Back-Error-Propagation parameters: $\alpha = 0.2$ and $\eta = 0.9$.



(a) Generalizing open loop configuration (parameter estimator).

(b) Generalizing closed loop configuration (RTAC); normalized control force.

(c) Generalizing closed loop configuration (RTAC); cart position.

(d) Generalizing closed loop configuration (RTAC); pendulum angle.

Figure 4. - Non-linear control law without noise. Initial state vector, $\vec{Z}(0)$ = (3m, 0, -35 °, 0). Normalization constant, F_g = 120 Newtons Back-Error-Propagation parameters: $\alpha = 0.2$ and η =0.9.

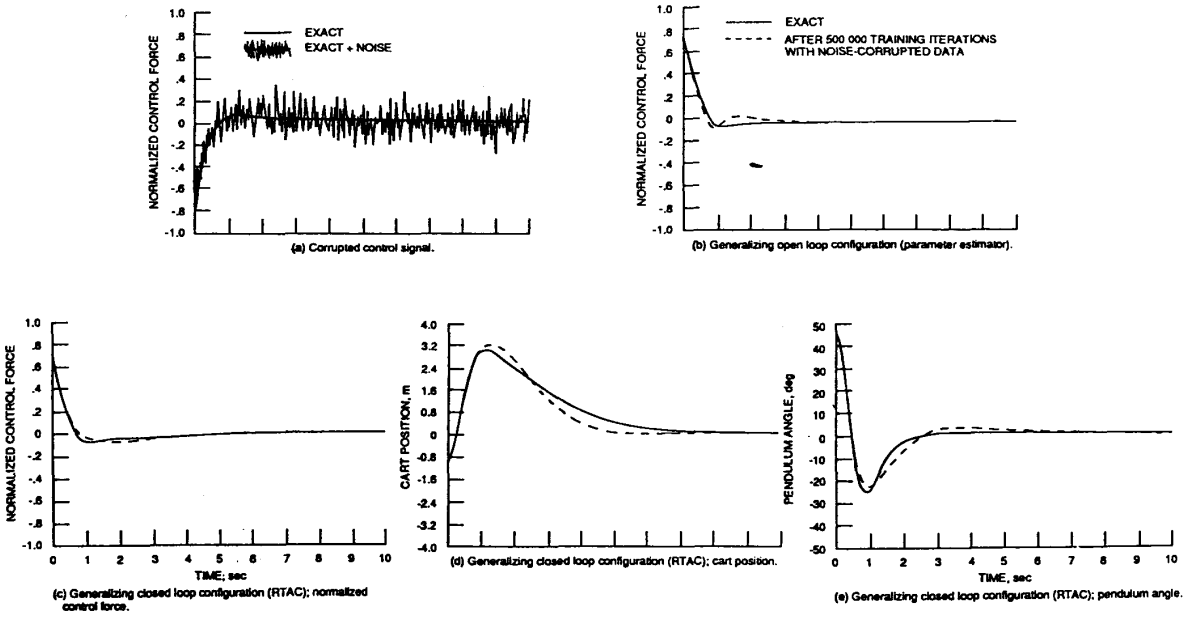


Figure 5. - Non-linear control law with noise for noise-to-signal ratio, N/S = 0.1; Normalization constant, F_g = 120 Newtons; Back-Error-Propagation parameters: $\alpha = 0.02$ and $\eta = 0$. Neural Net performance tested over noiseless data after training with noise-corrupted data.

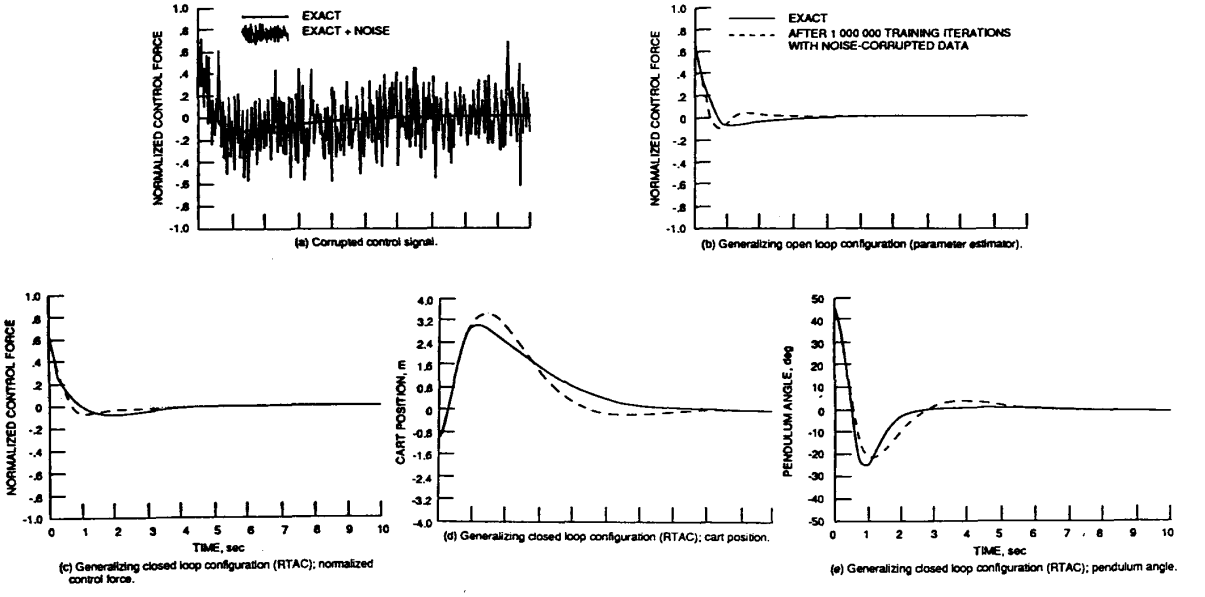


Figure 6. - Non-linear control law with noise for noise-to-signal ratio, N/S = 0.2; Normalization constant, F_g = 120 Newtons; Back-Error-Propagation parameters: $\alpha = 0.01$ and $\eta = 0$. Neural Net performance tested over noiseless data after training with noise-corrupted data.